Digital Connection

Don Rotolo, N2IRZ

High-speed packet making a comeback

This document ©2020 by Don Rotolo N2IRZ under CC BY-NC 4.0. To view a copy of this license, visit https://creativecommons.org/licenses/by-nc/4.0/.
This document was donated to the Digital Library of Amateur Radio & Communications (DLARC) for public access. It differs slightly from what appeared in *CQ Amateur Radio Magazine*.

Originally in CQ Magazine, January 2020

Happy New Year 2020, I hope this year is better for you than any other year so far.

My first experiences with digital radio were in the early years of packet radio. Tucson Amateur Packet Radio (TAPR, <www.tapr.org>) had just come out with the Z80-based TNC-2 terminal node controller, the network protocol wars had not yet heated up, and CQ Amateur Radio magazine wasn't even 35 yet. What interested me the most was how running a relatively slow digital data network could teach you how networks work, which was something I found valuable in my professional life. Other aspects of networking, such as antenna selection, fade margins, packet routing and site selection were just icing on the cake.

As packet radio matured, we saw several innovations, particularly in the modem (MOdulator-DEModulator), the hardware that produces and decodes the actual audio signals to and from the radio. Most packet links used the Bell 202 modem, developed for use with telephone systems, typically at 1200 baud, but with a minor tweak could be made to work at 2400 baud. We then saw the G3RUH modem (and several clones) for 9600, 19200 baud and higher, the Hamilton Area Packet Network's HAPN-T at 4800 baud (which works well on phase-modulated radios), and even the GRAPES modem at an astounding 56k baud, to name but a few. For these higher-speed modems, the biggest problem was finding a radio that could handle the signals without distortion.

Alas, finding radios that have the characteristics to support high-speed data remains difficult today. There are some commercial radios optimized – or at least usable – for high-speed data, with several offerings in the used market. These have their own issues, and testing shows that not every "9600 Baud Ready" radio really is.

This Month

This month, we will start a short series of articles on high(er) speed data. As promised in November's issue, we'll touch on the new 9600 baud KISS TNC developed by Nino Carrillo KK4HEJ. But instead of a deep dive into all the details, this month we will explore details of the hardware and firmware design, and in March, what to look for in a radio, and get into practical aspects of connections and making this actually work on the air.

The first test batch of Nino's TNCs went out in August 2019, and based on practical testing (see Figure 1) several design changes were recommended. Since at the time of this writing (November 2019) these changes are not yet completed, I decided that instead of introducing hardware that was going to be the subject of considerable changes, the focus would be on design.

Figure 1: A prototype NinoTNC, installed in a plastic case, as part of a test link at Tadd Torborg KA2DEW's house. The link managed over 300 Bytes per second. Photo courtesy Tadd Torborg KA2DEW.

Packet Comes Back

Long after the death of packet radio from internet poisoning, packet radio networking continues to grow (see <www.tarpn.org>), and some networks have links that are starting to approach saturation at 1200 baud. One easy solution to this problem is to swap in a higher-speed link.

I've written about the TARPN system a few times previously, as this represents in my opinion the ideal way to go about packet networking: Only a few simple rules, including mandatory use of dedicated point-to-point links, and anyone who wants to use the network has to have some skin in the game. This makes the network both robust and sustainable. The network was designed to be both transparent in operation, and highly inspectable, making it very easy to see how things work 'behind the scenes' – in other words, as an educational tool that is also valuable as a data network.

When Nino announced his KISS TNC for 9600 baud on the TARPN user group <https://groups.io/g/tarpn> last August, I was immediately interested. Since then, several test units have been in the hands of a few TARPN operators and the project is now being redesigned for its second iteration of circuit boards. Nino reported end-to-end transfer rates of about 600 bytes per second under ideal conditions, quite a lot faster than a good 1200 baud link.

Of course, the performance of this (or any) modem very much depends on having a proper RF path and configuration. If you set up a 9600 baud link that suffers from Hidden Transmitter Syndrome, throughput can easily fall below that of a 1200 baud link. A dedicated point-to-point link with sufficient signal strength is needed to have any chance of getting the performance you are paying for in the link.

Intended for use with a TARPN node controller running on a Raspberry Pi microcomputer, the KISS interface offers flexibility to use the TNC in almost any packet system, including stand-alone as a user terminal.

A Redesign

The issues found with the first batch of hardware were primarily things like physical compatibility for mounting and non-standard electrical connections. So the board layout is being changed to have the mounting holes match that of the TNC-PI from Coastal Chipworks < http://www.coastalchip.com/>, making it easier to mount in a standard TARPN box, and the addition of a DE-9 connector to be again compatible with the TNC-PI. It seems this is becoming a de-facto standard, even though this configuration has been around for a long time.

Figure 2: A DE-9 Female connector, the de-facto standard to interface a TNC with a data radio. My first experience with this configuration was on the TEKK KS-900 data radios. Pin 1 is data to the radio, Pin 2 is ground, Pin 4 is PTT and Pin 5 is data from the radio. Wikimedia photo

Some differences include the omission of I2C bus support – not a priority at the moment – and its ribbon cable, in favor of a USB connection to the Raspberry Pi using a mini-USB on the TNC. The TXDelay potentiometer is removed in favor of a software-only setting. (The TNC-PI can use either software or the on-board potentiometer R6 to adjust TXDelay). And, particularly valuable for in the development stages of most any project, additional status LEDs. (My personal favorite is the one named "Stranger" – see below).

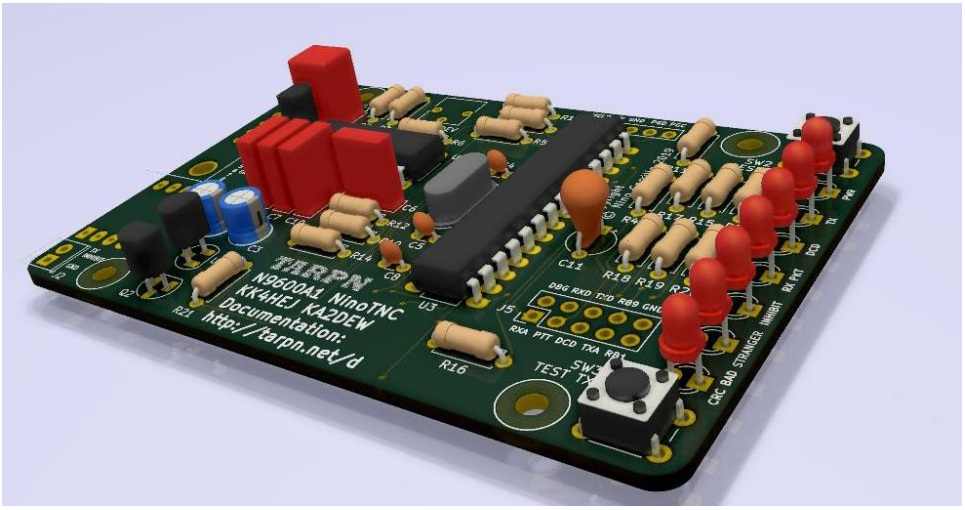

Figure 3: A rendering of the NinoTNC which is close to what's being designed. Not installed are the connectors for the host Raspberry Pi and the radio interface, a DE-9F connector. The heart of the system is the dsPIC IC seen in the center. Render courtesy of Tadd Torborg KA2DEW and Nino Carrillo KK4HEJ.

The goal here is to make everything both easily reproducible and turn-key. For example, even thought the GRAPES 56k modem was an

astounding achievement for its time, not many examples were actually placed into service. The main reason for that was the requirement that a transverter be used for the RF stage, made necessary by the lack of radios that could actually transmit and receive a signal that wide. So the GRAPES modem was designed with its RF output at around 30 MHz, and a transverter (with its naturally wide bandwidth) was used to up-convert the signal to a band where 56k was both useful and legal.

Figure 4: A 56k GRAPES modem. Note that it is called an RF modem: The output was not at baseband (audio) but at around 30 MHz, requiring a transverter to place the signal on a legal frequency. This relative complexity was the primary challenge to its widespread use. Photo courtesy Dale Heatherington WA4DSY.

The lesson learned from this and several other examples was that if the user needs to modify something to make it work, acceptance and usage will suffer significantly. So far, a TARPN node is completely off-the-shelf turn-key, with the exception of trivial things like the cable connecting the TNC to the radio and putting up antennas. Oh, and maybe the wooden box in which to mount a node, but it's just a convenience thing, not really required.

9600 Baud Radios

With this new TNC, a few hurdles remain to the entire system being turn-key, most importantly the selection of and connection to a compatible radio. You see, testing shows that several modern radios marketed and advertised as 9600 baud ready, aren't. The details of this will be covered in another month, as it is too large to fit here today.

Hardware

The fundamentals of the design aren't changing much, so let's take a good look at what we have. Before we start: If there is something you don't recognize or understand, take a moment to look on the Internet. You will find yourself understanding all this in greater detail, which will greatly increase your appreciation and satisfaction with what we are discussing. I just don't have the space to explain every detail, sorry, but I'll give it a try.

Nino's design uses a dsPIC from Microchip <www.microchip.com>, which is a 16-bit PIC microcontroller that incorporates DSP (digital signal processing) functionality and features low power consumption. The circuit board uses through-hole components to allow for easier construction.

The modulated data for the transmit (TX) chain out to the radio comes from the output of the dsPIC's DAC (digital-to-analog converter), which is filtered using a low-pass 4th order Bessel filter implemented in two op-amps to produce a clean waveform. A potentiometer is inserted into the signal path to allow the signal level to the radio to be adjusted, which sets the transmit deviation. As most anyone working with digital signals knows, excessive deviation seriously degrades the signal quality, dramatically reducing the ability of the receiver to decode incoming data.

On the receive side, a simple high-pass filter, with an op-amp buffer, centers the incoming audio stream in the dsPIC's ADC (analog-to-digital converter). What this means is that the input range of the ADC is limited, and this circuit scales the incoming audio to ensure nothing gets overloaded or out-of-range. This allows the design to handle a wide range of receive audio input levels. The ADC then converts the audio to a digital bit stream for processing and decoding.

Figure 5: A HAPN-T 4800 baud modem atop my Icom IC-45A. To operate at 4800 baud, the radio required a minor modification to tap into the discriminator output and modulator input, which are brought out (along with PTT) on the DE-9F connector shown. Such modifications scared many packeteers away from anything faster than 1200 baud years

PTT (push-to-talk, switching on the transmitter) is handled by a simple 2N700 FET (field-effect transistor) circuit. The circuit also contains a simple power supply to convert the main 5 volt supply to 3.3 volts for the dsPIC and serial data systems. In the first prototype, screw terminals were used for the radio connections, but in the redesign this will be a DE-9 connector in the standard configuration. There is a test button to generate a tone to help with setting deviation. A few LEDs are included to help the user understand system status and operation, and as I mentioned above the redesign will add a few more.

Firmware

The dsPIC is used to packetize the KISS data from a host – the Raspberry Pi in the case of a TARPN node, but it doesn't have to be that – into standard AX.25 format, including bit-stuffing (a way of ensuring the digital signal doesn't have too many 1s or 0s in a row), CRC (cyclic redundancy check, a way of helping ensure data integrity) and scrambling (to make the signal seem more random, making transmitting and receiving more robust). The output waveform is then generated using the dsPIC's output compare module as a 9-bit PWM (pulse-width modulated) DAC running at nine times the baud rate. The waveform is smoothed with a Gaussian FIR pulse-shaping filter, and this output from the dsPIC is fed to the previously mentioned low-pass Bessel filter to integrate the PWM pulses, forming a smooth analog waveform which is fed to the transmitter. The transmit signal level adjustment potentiometer is buffered by an op amp in the transmit chain.

For reception, the filtered and level-adjusted receive analog audio is converted into a digital bit stream by the dsPIC's ADC, sampling at five times the baud rate. Recovery of the clock (and data) is implemented in the dsPIC using peak-detection and zero-crossing detection. DCD (data carrier detection, important for running open-squelch) is performed by evaluating the clock sync. The data is un-scrambled, decoded, bit stuffing is removed and CRC validated. If all is well, the data is forwarded to the host microcomputer either over the USB port or the serial port. As mentioned previously, unlike the TNC-Pi, the I2C serial link simply isn't fast enough to be used for this.

The TNC as implemented understands all standard KISSPARM settings (such as TXDelay and PACLen). The firmware does not implement the transmit deviation adjustment, even though it could, only because the adjustment is far easier and more intuitive with a screwdriver and trimmer potentiometer. Pressing the "test" button causes the TNC to transmit a 998 Hz tone so the deviation can be set. 998 Hz is used because it allows setting the perfect minimum-shift deviation by finding the first Bessel filter null, which occurs at a point where the tone frequency is multiplied by 2.405, which is 2400 Hz.

Setting the deviation correctly is important. A 2400 Hz deviation is ideal, and while a somewhat smaller deviation is not terribly harmful, a somewhat larger deviation will introduce distortions into the transmit chain that will significantly degrade the link's performance. Of course, the ideal would be to have a deviation meter, which provides an objective number for the deviation. But there are several alternatives, which we'll discuss another month when we talk about connecting this to a radio.

In March, I expect the second generation of this project to be well-enough along to discuss the practical aspects of getting it on the air, along selecting a suitable radio. In the meantime, go check out the TARPN documentation page for this TNC at <tarpn.net/d>. I am also hopeful that plans to make these more widely available will be formed, but with that I can't make any promises. Remember the desire to make everything turn-key, and as with many of us, operating a business that builds ham radio TNCs and modems might interfere with our day job. We'll see.

Until then, 73 de N2IRZ.

2025 NOTE: The NinoTNC was very new when this article was written. It is mature now. See* http://tarpn.net *for more info.

###